Digital Negative (DNG) Specification

Version 1.3.0.0

June 2009

ADOBE SYSTEMS INCORPORATED

Corporate Headquarters
345 Park Avenue
San Jose, CA 95110-2704
(408) 536-6000
http://www.adobe.com

Table of Contents

Preface

About This Document

The Digital Negative (DNG) Specification describes a non-proprietary file format for storing camera raw files that can be used by a wide range of hardware and software vendors.

This section contains information about this document, including how it is organized and where to go for additional information.

Audience

This document is intended for developers of hardware and software applications that will generate, process, manage, or archive camera raw files.

How This Document Is Organized

This document has the following sections:

- Chapter 1, “Introduction” explains what digital negatives are, gives an overview of the DNG file format, and discusses the advantages of DNG.
- Chapter 2, “DNG Format Overview” provides an overview of the DNG format, including information on file extensions, SubIFD trees, byte order, masked pixels, defective pixels, metadata, and proprietary data.
- Chapter 3, “Restrictions and Extensions to Existing TIFF Tags” describes tag differences between DNG and the TIFF 6.0 format on which DNG is based.
- Chapter 4, “DNG Tags” lists all DNG-specific tags and describes how they are used.
- Chapter 5, “Mapping Raw Values to Linear Reference Values” specifies DNG's processing model for mapping stored raw sensor values into linear reference values.
- Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” describes DNG's processing model for mapping between the camera color space coordinates (linear reference values) and CIE XYZ coordinates.
- Chapter 7, “Opcode List Processing” describes the concept of “Opcode Lists”, which allow additional processing steps to be specified in an extensible manner.
- “Appendix A: Compatibility with Previous Versions” documents compatibility between the current and previous DNG versions.

New Information for Version 1.3.0.0

The following changes have been made for the 1.3.0.0 version of this specification:

In Chapter 2, “DNG Format Overview”, the section on Defective Pixels has been updated.

The section on Opcode Lists has been added to Chapter 2, “DNG Format Overview”.

The CFALayout tag has been updated in Chapter 4, “DNG Tags”.

Additional Tags for Version 1.3.0.0 were added to Chapter 4, “DNG Tags”.

Chapter 7, “Opcode List Processing” was added.

“Appendix A: Compatibility with Previous Versions” was updated.

Where to Go for More Information

DNG is an extension of TIFF 6.0 and is compatible with the TIFF-EP standard. See these specifications for more information on TIFF and TIFF-EP:

TIFF 6.0 Specification, Adobe Systems, Inc., 1992-06-03.	http://partners.adobe.com/asn/developer/pdfs/tn/TIFF6.pdf
TIFF/EP Specification, ISO/DIS 12234-2, 2001-10-15.	http://www.iso.org

1 Introduction

The Pros and Cons of Raw Data

Seeking a greater degree of flexibility and artistic control, professional photographers increasingly opt to manipulate raw data from their digital cameras. Unlike JPEG and TIFF formats which store images that have been processed by the camera, camera raw files capture unprocessed or minimally processed data directly from the camera sensor. Because they are analogous to film negatives in a photographer's workflow, camera raw formats are often referred to as "digital negatives."

Camera raw formats offer both advantages and disadvantages. One advantage is increased artistic control for the end user. The user can precisely adjust a range of parameters, including white balance, tone mapping, noise reduction, sharpening and others, to achieve a desired look.

One disadvantage is that unlike JPEG and TIFF files which are ready for immediate use, camera raw files must be processed before they can be used, typically through software provided by the camera manufacturer or through a converter like the Camera Raw plug-in for Adobe Photoshop® software.

The challenge for end users and camera vendors alike is that there is no publicly-documented and supported format for storing raw camera data. Every camera manufacturer that supports raw data must create their own proprietary format, along with software for converting the proprietary format into the standard JPEG and/or TIFF formats.

A Standard Format

The lack of a standard format for camera raw files creates additional work for camera manufacturers because they need to develop proprietary formats along with the software to process them. It also poses risks for end users. Camera raw formats vary from camera to camera, even those produced by the same manufacturer. It is not uncommon for a camera manufacturer to terminate support for a discontinued camera's raw format. This means users have no guarantee they will be able to open archived camera raw files in the future.

To address these problems, Adobe has defined a new non-proprietary format for camera raw files. The format, called Digital Negative or DNG, can be used by a wide range of hardware and software developers to provide a more flexible raw processing and archiving workflow. End users may also use DNG as an intermediate format for storing images that were originally captured using a proprietary camera raw format.

The Advantages of DNG

DNG has all the benefits of current camera raw formats; namely, increased flexibility and artistic control. In addition, DNG offers several new advantages over proprietary camera raw formats.

Self-Contained

With the current proprietary camera raw formats, software programs wishing to process camera raw files must have specific information about the camera that created the file. As new camera models are released, software manufacturers (and by extension users) must update their software to accommodate the new camera raw formats.

Because DNG metadata is publicly documented, software readers such as the Adobe Photoshop Camera Raw plug-in do not need camera-specific knowledge to decode and process files created by a camera that supports DNG. That means reduced software maintenance and a more self-contained solution for end users.

Archival

Camera manufacturers sometimes drop support for a propriety raw format a few years after a camera is discontinued. Without continued software support, users may not be able to access images stored in proprietary raw formats and the images may be lost forever. Since DNG is publicly documented, it is far more likely that raw images stored as DNG files will be readable by software in the distant future, making DNG a safer choice for archival.

TIFF Compatible

DNG is an extension of the TIFF 6.0 format, and is compatible with the TIFF-EP standard. It is possible (but not required) for a DNG file to simultaneously comply with both the Digital Negative specification and the TIFF-EP standard.

2 DNG Format Overview

This section describes the DNG format. As an extension of the TIFF 6.0 format, DNG should follow all the formatting rules for TIFF 6.0. For more information, refer to the TIFF 6.0 specification.

The following topics are discussed in this section:

- File Extensions
- SubIFD Trees
- Byte Order
- Masked Pixels
- Defective Pixels
- Metadata
- Proprietary Data
- Camera Profiles
- Opcode Lists

File Extensions

The recommended file extension for Digital Negative is ".DNG". Readers should accept either the ".DNG" or ".TIF" extensions for compatibility with TIFF-EP.

SubIFD Trees

DNG recommends the use of SubIFD trees, as described in the TIFF-EP specification. SubIFD chains are not supported.

The highest-resolution and quality IFD should use NewSubFileType equal to 0. Reduced resolution (or quality) thumbnails or previews, if any, should use NewSubFileType equal to 1 (for a primary preview) or 10001.H (for an alternate preview).

DNG recommends, but does not require, that the first IFD contain a low-resolution thumbnail, as described in the TIFF-EP specification.

Byte Order

DNG readers are required to support either byte order, even for files from a particular camera model. Writers can write either byte order, whichever is easier and/or faster for the writer.

Masked Pixels

Most camera sensors measure the black encoding level using fully-masked pixels at the edges of the sensor. These pixels can either be trimmed before storing the image in DNG, or they can be included in the stored image. If the masked pixels are not trimmed, the area of the non-masked pixels must be specified using the ActiveArea tag.

The black encoding level information extracted from these masked pixels should be used to either pre-compensate the raw data stored in the file or they should be included in the file using the DNG tags for specifying the black level.

This black encoding level information is required even if the masked pixels are not trimmed, to allow DNG readers to process the image without requiring knowledge of the best way to compute the black levels for any given camera model.

Defective Pixels

There are two ways to deal with defective pixels in DNG. The first is to map out (interpolate over) the defective pixels before storing the raw data in DNG. The second is to include a bad pixel fixing opcode in the OpcodeList1 tag.

Metadata

Additional metadata may be embedded in DNG in the following ways:

- Using TIFF-EP or EXIF metadata tags
- Using the IPTC metadata tag (33723)
- Using the XMP metadata tag (700)

Note that TIFF-EP and EXIF use nearly the same metadata tag set, but TIFF-EP stores the tags in IFD 0, while EXIF store the tags in a separate IFD. Either location is allowed by DNG, but the EXIF location is preferred.

Proprietary Data

Camera manufacturers may want to include proprietary data in a raw file for use by their own raw converter. DNG allows proprietary data to be stored using private tags, private IFDs, and/or a private MakerNote.

It is recommended that manufacturers use the DNGPrivateData and MakerNoteSafety tags to ensure that programs that edit DNG files preserve this proprietary data. See Chapter 4, “DNG Tags” on page 17 for more information on the DNGPrivateData and MakerNoteSafety tags.

Camera Profiles

DNG 1.2.0.0 and later formalizes the concept of a “camera profile” and allows multiple camera profiles to be embedded in a single DNG file. A camera profile consists of a set of tags (both existing in DNG versions earlier than 1.2.0.0 and newly defined in DNG version 1.2.0.0), some of which are optional.

The set of tags belonging to a camera profile includes the following:

- ColorMatrix1
- ColorMatrix2
- ReductionMatrix1
- ReductionMatrix2
- CalibrationIlluminant1
- CalibrationIlluminant2
- ProfileCalibrationSignature
- ProfileName
- ProfileHueSatMapDims
- ProfileHueSatMapData1
- ProfileHueSatMapData2
- ProfileToneCurve
- ProfileEmbedPolicy
- ProfileCopyright
- ForwardMatrix1
- ForwardMatrix1
- ProfileLookTableDims
- ProfileLookTableData

The primary camera profile is stored in IFD 0, as it was for DNG versions earlier than 1.2.0.0. This allows backward compatibility with older DNG readers. DNG allows additional camera profiles to be embedded using the ExtraCameraProfiles tag, which points to a list of Camera Profile IFDs.

Opcode Lists

DNG 1.3.0.0 and later incorporates the concept of "Opcode Lists", which allow additional processing steps to be specified in an extensible manner.

This allows complex processing to be moved off the camera hardware, which often has limited processing power, and into the DNG reader, which is often running on more powerful hardware.

This also allows processing steps to be specified, such as lens corrections, which ideally should be performed on the image data after it has been demosaiced, while still retaining the advantages of a raw mosaic data format.

The set of tags controlling this feature are:

- OpcodeList1
- OpcodeList2
- OpcodeList3

3 Restrictions and Extensions to Existing TIFF Tags

This section describes the restrictions and extension to the following TIFF tags:

- NewSubFileType
- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation

NewSubFileType

In DNG versions earlier than 1.2.0.0, full resolution raw images should use NewSubFileType equal to 0. Rendered previews or reduced resolution versions of raw images should use NewSubFileType equal to 1. DNG 1.2.0.0 allows a new value for NewSubFileType equal to 10001.H. This value, used for alternative or non-primary rendered previews, allows for multiple renderings (not just multiple sizes of a single rendering) to be stored in a DNG file. DNG reading software that displays a preview for a DNG file should, by default, display a preview from an IFD with NewSubFileType equal to 1. Alternative renderings should only be displayed if requested by the user.

BitsPerSample

Supported values are from 8 to 32 bits/sample. The depth must be the same for each sample if SamplesPerPixel is not equal to 1. If BitsPerSample is not equal to 8 or 16 or 32, then the bits must be packed into bytes using the TIFF default FillOrder of 1 (big-endian), even if the TIFF file itself uses little-endian byte order.

Compression

Two Compression tag values are supported:

- Value = 1: Uncompressed data.
- Value = 7: JPEG compressed data, either baseline DCT JPEG, or lossless JPEG compression.

If PhotometricInterpretation = 6 (YCbCr) and BitsPerSample = 8/8/8, or if PhotometricInterpretation = 1 (BlackIsZero) and BitsPerSample = 8, then the JPEG variant must be baseline DCT JPEG.

Otherwise, the JPEG variant must be lossless Huffman JPEG. For lossless JPEG, the internal width/length/components in the JPEG stream are not required to match the strip or tile's width/length/components. Only the total sample counts need to match. It is common for CFA images to be encoded with a different width, length or component count to allow the JPEG compression predictors to work across like colors.

PhotometricInterpretation

The following values are supported for thumbnail and preview IFDs only:

- 1 = BlackIsZero. Assumed to be in a gamma 2.2 color space, unless otherwise specified using PreviewColorSpace tag.
- 2 = RGB. Assumed to be in the sRGB color space, unless otherwise specified using the PreviewColorSpace tag.
- 6 = YCbCr. Used for JPEG encoded preview images.

The following values are supported for the raw IFD, and are assumed to be the camera's native color space:

- 32803 = CFA (Color Filter Array).
- 34892 = LinearRaw.

The CFA PhotometricInterpretation value is documented in the TIFF-EP specification. Its use requires the use of the CFARepeatPatternDim and CFAPattern tags in the same IFD. The origin of the repeating CFA pattern is the top-left corner of the ActiveArea rectangle.

The LinearRaw PhotometricInterpretation value is intended for use by cameras that do not use color filter arrays, but instead capture all color components at each pixel. It can also be used for CFA data that has already been de-mosaiced.

The LinearRaw value can be used in reduced resolution IFDs, even if the raw IFD uses the CFA PhotometricInterpretation value.

Orientation

Orientation is a required tag for DNG. With the Orientation tag present, file browsers can perform lossless rotation of DNG files by modifying a single byte of the file. DNG readers should support all possible orientations, including mirrored orientations. Note that the mirrored orientations are not allowed by the TIFF-EP specification, so writers should not use them if they want their files be compatible with both specifications.

4 DNG Tags

This section describes DNG-specific tags. Note that the tags listed here are not part of the TIFF-EP specification.

DNGVersion

Tag	50706 (C612.H)
Type	BYTE
Count	4
Value	See below
Default	Required tag
Usage	IFD 0

Description

This tag encodes the DNG four-tier version number. For files compliant with this version of the DNG specification (1.2.0.0), this tag should contain the bytes: 1, 2, 0, 0.

DNGBackwardVersion

Tag	50707 (C613.H)
Type	BYTE
Count	4
Value	See below
Default	DNGVersion with the last two bytes set to zero.
Usage	IFD 0

Description

This tag specifies the oldest version of the Digital Negative specification for which a file is compatible. Readers should not attempt to read a file if this tag specifies a version number that is higher than the version number of the specification the reader was based on.

In addition to checking the version tags, readers should, for all tags, check the types, counts, and values, to verify it is able to correctly read the file.

For more information on compatibility with previous DNG versions, see Appendix A: Compatibility with Previous Versions.

UniqueCameraModel

Tag	50708 (C614.H)
Type	ASCII
Count	String length including null
Value	Null terminated string
Default	Required tag
Usage	IFD 0

Description

UniqueCameraModel defines a unique, non-localized name for the camera model that created the image in the raw file. This name should include the manufacturer's name to avoid conflicts, and should not be localized, even if the camera name itself is localized for different markets (see LocalizedCameraModel).

This string may be used by reader software to index into per-model preferences and replacement profiles.

Examples of unique model names are:

- "Canon EOS 300D"
- "Fujifilm FinePix S2Pro"
- "Kodak ProBack645"
- "Minolta DiMAGE A1"
- "Nikon D1X"
- "Olympus C-5050Z"
- "Pentax *istD"
- "Sony F828"

LocalizedCameraModel

Tag	50709 (C615.H)
Type	ASCII or BYTE
Count	Byte count including null
Value	Null terminated UTF-8 encoded Unicode string
Default	Same as UniqueCameraModel
Usage	IFD 0

Description

Similar to the UniqueCameraModel field, except the name can be localized for different markets to match the localization of the camera name.

CFAPlaneColor

Tag	50710 (C616.H)
Type	BYTE
Count	ColorPlanes
Value	See below
Default	0, 1, 2 (red, green, blue)
Usage	Raw IFD

Description

CFAPlaneColor provides a mapping between the values in the CFAPattern tag and the plane numbers in LinearRaw space. This is a required tag for non-RGB CFA images.

CFALayout

Tag	50711 (C617.H)
Type	SHORT
Count	1
Value	See below
Default	1
Usage	Raw IFD

Description

CFALayout describes the spatial layout of the CFA. The currently defined values are:

1 = Rectangular (or square) layout

2 = Staggered layout A: even columns are offset down by 1/2 row

3 = Staggered layout B: even columns are offset up by 1/2 row

4 = Staggered layout C: even rows are offset right by 1/2 column

5 = Staggered layout D: even rows are offset left by 1/2 column

6 = Staggered layout E: even rows are offset up by 1/2 row, even columns are offset left by 1/2 column

7 = Staggered layout F: even rows are offset up by 1/2 row, even columns are offset right by 1/2 column

8 = Staggered layout G: even rows are offset down by 1/2 row, even columns are offset left by 1/2 column

9 = Staggered layout H: even rows are offset down by 1/2 row, even columns are offset right by 1/2 column

Note that for the purposes of this tag, rows and columns are numbered starting with one.

Layout values 6 through 9 were added with DNG version 1.3.0.0.

LinearizationTable

Tag	50712 (C618.H)
Type	SHORT
Count	N
Value	See below
Default	Identity table (0, 1, 2, 3, etc.)
Usage	Raw IFD

Description

LinearizationTable describes a lookup table that maps stored values into linear values. This tag is typically used to increase compression ratios by storing the raw data in a non-linear, more visually uniform space with fewer total encoding levels.

If SamplesPerPixel is not equal to one, this single table applies to all the samples for each pixel.

See Chapter 5, “Mapping Raw Values to Linear Reference Values” on page 65 for details of the processing model.

BlackLevelRepeatDim

Tag	50713 (C619.H)
Type	SHORT
Count	2
Value	Value 0: BlackLevelRepeatRows Value 1: BlackLevelRepeatCols
Default	1, 1
Usage	Raw IFD

Description

This tag specifies repeat pattern size for the BlackLevel tag.

BlackLevel

Tag	50714 (C61A.H)
Type	SHORT or LONG or RATIONAL
Count	BlackLevelRepeatRows * BlackLevelRepeatCols * SamplesPerPixel
Value	See below
Default	0
Usage	Raw IFD

Description

This tag specifies the zero light (a.k.a. thermal black or black current) encoding level, as a repeating pattern. The origin of this pattern is the top-left corner of the ActiveArea rectangle. The values are stored in row-column-sample scan order.

See Chapter 5, “Mapping Raw Values to Linear Reference Values” on page 65 for details of the processing model.

BlackLevelDeltaH

Tag	50715 (C61B.H)
Type	SRATIONAL
Count	ActiveArea width
Value	See below
Default	All zeros
Usage	Raw IFD

Description

If the zero light encoding level is a function of the image column, BlackLevelDeltaH specifies the difference between the zero light encoding level for each column and the baseline zero light encoding level.

If SamplesPerPixel is not equal to one, this single table applies to all the samples for each pixel.

See Chapter 5, “Mapping Raw Values to Linear Reference Values” on page 65 for details of the processing model.

BlackLevelDeltaV

Tag	50716 (C61C.H)
Type	SRATIONAL
Count	ActiveArea length
Value	See below
Default	All zeros
Usage	Raw IFD

Description

If the zero light encoding level is a function of the image row, this tag specifies the difference between the zero light encoding level for each row and the baseline zero light encoding level.

If SamplesPerPixel is not equal to one, this single table applies to all the samples for each pixel.

See Chapter 5, “Mapping Raw Values to Linear Reference Values” on page 65 for details of the processing model.

WhiteLevel

Tag	50717 (C61D.H)
Type	SHORT or LONG
Count	SamplesPerPixel
Value	See below
Default	(2 ** BitsPerSample) - 1
Usage	Raw IFD

Description

This tag specifies the fully saturated encoding level for the raw sample values. Saturation is caused either by the sensor itself becoming highly non-linear in response, or by the camera's analog to digital converter clipping.

See Chapter 5, “Mapping Raw Values to Linear Reference Values” on page 65 for details of the processing model.

DefaultScale

Tag	50718 (C61E.H)
Type	RATIONAL
Count	2
Value	Value 0: DefaultScaleH Value 1: DefaultScaleV
Default	1.0, 1.0
Usage	Raw IFD

Description

DefaultScale is required for cameras with non-square pixels. It specifies the default scale factors for each direction to convert the image to square pixels. Typically these factors are selected to approximately preserve total pixel count.

For CFA images that use CFALayout equal to 2, 3, 4, or 5, such as the Fujifilm SuperCCD, these two values should usually differ by a factor of 2.0.

BestQualityScale

Tag	50780 (C65C.H)
Type	RATIONAL
Count	1
Value	see below
Default	1.0
Usage	Raw IFD

Description

For some cameras, the best possible image quality is not achieved by preserving the total pixel count during conversion. For example, Fujifilm SuperCCD images have maximum detail when their total pixel count is doubled.

This tag specifies the amount by which the values of the DefaultScale tag need to be multiplied to achieve the best quality image size.

DefaultCropOrigin

Tag	50719 (C61F.H)
Type	SHORT or LONG or RATIONAL
Count	2
Value	Value 0: DefaultCropOriginH Value 1: DefaultCropOriginV
Default	0, 0
Usage	Raw IFD

Description

Raw images often store extra pixels around the edges of the final image. These extra pixels help prevent interpolation artifacts near the edges of the final image.

DefaultCropOrigin specifies the origin of the final image area, in raw image coordinates (i.e., before the DefaultScale has been applied), relative to the top-left corner of the ActiveArea rectangle.

DefaultCropSize

Tag	50720 (C620.H)
Type	SHORT or LONG or RATIONAL
Count	2
Value	Value 0: DefaultCropSizeH Value 1: DefaultCropSizeV
Default	ImageWidth, ImageLength
Usage	Raw IFD

Description

Raw images often store extra pixels around the edges of the final image. These extra pixels help prevent interpolation artifacts near the edges of the final image.

DefaultCropSize specifies the size of the final image area, in raw image coordinates (i.e., before the DefaultScale has been applied).

CalibrationIlluminant1

Tag	50778 (C65A.H)
Type	SHORT
Count	1
Value	See below
Default	0 (unknown)
Usage	IFD 0 or Camera Profile IFD

Description

The illuminant used for the first set of color calibration tags. The legal values for this tag are the same as the legal values for the LightSource EXIF tag.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

CalibrationIlluminant2

Tag	50779 (C65B.H)
Type	SHORT
Count	1
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

The illuminant used for an optional second set of color calibration tags. The legal values for this tag are the same as the legal values for the CalibrationIlluminant1 tag; however, if both are included, neither is allowed to have a value of 0 (unknown).

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

ColorMatrix1

Tag	50721 (C621.H)
Type	SRATIONAL
Count	ColorPlanes * 3
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

ColorMatrix1 defines a transformation matrix that converts XYZ values to reference camera native color space values, under the first calibration illuminant. The matrix values are stored in row scan order.

The ColorMatrix1 tag is required for all non-monochrome DNG files.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

ColorMatrix2

Tag	50722 (C622.H)
Type	SRATIONAL
Count	ColorPlanes * 3
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

ColorMatrix2 defines a transformation matrix that converts XYZ values to reference camera native color space values, under the second calibration illuminant. The matrix values are stored in row scan order.

See Chapter 6, "Mapping Camera Color Space to CIE XYZ Space" on page 67 for details of the color-processing model.

CameraCalibration1

Tag	50723 (C623.H)
Type	SRATIONAL
Count	ColorPlanes * ColorPlanes
Value	See below
Default	Identity matrix
Usage	IFD 0

Description

CameraCalibration1 defines a calibration matrix that transforms reference camera native space values to individual camera native space values under the first calibration illuminant. The matrix is stored in row scan order.

This matrix is stored separately from the matrix specified by the ColorMatrix1 tag to allow raw converters to swap in replacement color matrices based on UniqueCameraModel tag, while still taking advantage of any per-individual camera calibration performed by the camera manufacturer.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

CameraCalibration2

Tag	50724 (C624.H)
Type	SRATIONAL
Count	ColorPlanes * ColorPlanes
Value	See below
Default	Identity matrix
Usage	IFD 0

Description

CameraCalibration2 defines a calibration matrix that transforms reference camera native space values to individual camera native space values under the second calibration illuminant. The matrix is stored in row scan order.

This matrix is stored separately from the matrix specified by the ColorMatrix2 tag to allow raw converters to swap in replacement color matrices based on UniqueCameraModel tag, while still taking advantage of any per-individual camera calibration performed by the camera manufacturer.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

ReductionMatrix1

Tag	50725 (C625.H)
Type	SRATIONAL
Count	3 * ColorPlanes
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

ReductionMatrix1 defines a dimensionality reduction matrix for use as the first stage in converting color camera native space values to XYZ values, under the first calibration illuminant. This tag may only be used if ColorPlanes is greater than 3. The matrix is stored in row scan order.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

ReductionMatrix2

Tag	50726 (C626.H)
Type	SRATIONAL
Count	3 * ColorPlanes
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

ReductionMatrix2 defines a dimensionality reduction matrix for use as the first stage in converting color camera native space values to XYZ values, under the second calibration illuminant. This tag may only be used if ColorPlanes is greater than 3. The matrix is stored in row scan order.

See Chapter 6, “Mapping Camera Color Space to CIE XYZ Space” on page 67 for details of the color-processing model.

AnalogBalance

Tag	50727 (C627.H)
Type	RATIONAL
Count	ColorPlanes
Value	See below
Default	All 1.0
Usage	IFD 0

Description

Normally the stored raw values are not white balanced, since any digital white balancing will reduce the dynamic range of the final image if the user decides to later adjust the white balance; however, if camera hardware is capable of white balancing the color channels before the signal is digitized, it can improve the dynamic range of the final image.

AnalogBalance defines the gain, either analog (recommended) or digital (not recommended) that has been applied the stored raw values.

See Chapter 6, "Mapping Camera Color Space to CIE XYZ Space" on page 67 for details of the color-processing model.

AsShotNeutral

Tag	50728 (C628.H)
Type	SHORT or RATIONAL
Count	ColorPlanes
Value	See below
Default	None
Usage	IFD 0

Description

AsShotNeutral specifies the selected white balance at time of capture, encoded as the coordinates of a perfectly neutral color in linear reference space values. The inclusion of this tag precludes the inclusion of the AsShotWhiteXY tag.

AsShotWhiteXY

Tag	50729 (C629.H)
Type	RATIONAL
Count	2
Value	See below
Default	None
Usage	IFD 0

Description

AsShotWhiteXY specifies the selected white balance at time of capture, encoded as x-y chromaticity coordinates. The inclusion of this tag precludes the inclusion of the AsShotNeutral tag.

BaselineExposure

Tag	50730 (C62A.H)
Type	SRATIONAL
Count	1
Value	See below
Default	0.0
Usage	IFD 0

Description

Camera models vary in the trade-off they make between highlight headroom and shadow noise. Some leave a significant amount of highlight headroom during a normal exposure. This allows significant negative exposure compensation to be applied during raw conversion, but also means normal exposures will contain more shadow noise. Other models leave less headroom during normal exposures. This allows for less negative exposure compensation, but results in lower shadow noise for normal exposures.

Because of these differences, a raw converter needs to vary the zero point of its exposure compensation control from model to model. BaselineExposure specifies by how much (in EV units) to move the zero point. Positive values result in brighter default results, while negative values result in darker default results.

BaselineNoise

Tag	50731 (C62B.H)
Type	RATIONAL
Count	1
Value	See below
Default	1.0
Usage	IFD 0

Description

BaselineNoise specifies the relative noise level of the camera model at a baseline ISO value of 100, compared to a reference camera model.

Since noise levels tend to vary approximately with the square root of the ISO value, a raw converter can use this value, combined with the current ISO, to estimate the relative noise level of the current image.

BaselineSharpness

Tag	50732 (C62C.H)
Type	RATIONAL
Count	1
Value	See below
Default	1.0
Usage	IFD 0

Description

BaselineSharpness specifies the relative amount of sharpening required for this camera model, compared to a reference camera model. Camera models vary in the strengths of their anti-aliasing filters. Cameras with weak or no filters require less sharpening than cameras with strong anti-aliasing filters.

BayerGreenSplit

Tag	50733 (C62D.H)
Type	LONG
Count	1
Value	See below
Default	0
Usage	Raw IFD

Description

BayerGreenSplit only applies to CFA images using a Bayer pattern filter array. This tag specifies, in arbitrary units, how closely the values of the green pixels in the blue/green rows track the values of the green pixels in the red/green rows.

A value of zero means the two kinds of green pixels track closely, while a non-zero value means they sometimes diverge. The useful range for this tag is from 0 (no divergence) to about 5000 (quite large divergence).

LinearResponseLimit

Tag	50734 (C62E.H)
Type	RATIONAL
Count	1
Value	See below
Default	1.0
Usage	IFD 0

Description

Some sensors have an unpredictable non-linearity in their response as they near the upper limit of their encoding range. This non-linearity results in color shifts in the highlight areas of the resulting image unless the raw converter compensates for this effect.

LinearResponseLimit specifies the fraction of the encoding range above which the response may become significantly non-linear.

CameraSerialNumber

Tag	50735 (C62F.H)
Type	ASCII
Count	String length including null
Value	Null terminated string
Default	None
Usage	IFD 0

Description

CameraSerialNumber contains the serial number of the camera or camera body that captured the image.

LensInfo

Tag	50736 (C630.H)
Type	RATIONAL
Count	4
Value	Value 0: Minimum focal length in mm. Value 1: Maximum focal length in mm. Value 2: Minimum (maximum aperture) f-stop at minimum focal length. Value 3: Minimum (maximum aperture) f-stop at maximum focal length.
Default	none
Usage	IFD 0

Description

LensInfo contains information about the lens that captured the image. If the minimum f-stops are unknown, they should be encoded as 0/0.

ChromaBlurRadius

Tag	50737 (C631.H)
Type	RATIONAL
Count	1
Value	Chroma blur radius in pixels
Default	See below
Usage	Raw IFD

Description

ChromaBlurRadius provides a hint to the DNG reader about how much chroma blur should be applied to the image. If this tag is omitted, the reader will use its default amount of chroma blurring.

Normally this tag is only included for non-CFA images, since the amount of chroma blur required for mosaic images is highly dependent on the de-mosaic algorithm, in which case the DNG reader's default value is likely optimized for its particular de-mosaic algorithm.

AntiAliasStrength

Tag	50738 (C632.H)
Type	RATIONAL
Count	1
Value	Relative strength of the camera's anti-alias filter
Default	1.0
Usage	Raw IFD

Description

AntiAliasStrength provides a hint to the DNG reader about how strong the camera's anti-alias filter is. A value of 0.0 means no anti-alias filter (i.e., the camera is prone to aliasing artifacts with some subjects), while a value of 1.0 means a strong anti-alias filter (i.e., the camera almost never has aliasing artifacts).

Note that this tag overlaps in functionality with the BaselineSharpness tag. The primary difference is the AntiAliasStrength tag is used as a hint to the de-mosaic algorithm, while the BaselineSharpness tag is used as a hint to a sharpening algorithm applied later in the processing pipeline.

ShadowScale

Tag	50739 (C633.H)
Type	RATIONAL
Count	1
Value	See below
Default	1.0
Usage	IFD 0

Description

This tag is used by Adobe Camera Raw to control the sensitivity of its "Shadows" slider.

DNGPrivateData

Tag	50740 (C634.H)
Type	BYTE
Count	Length of private data block in bytes
Value	See below
Default	None
Usage	IFD 0

Description

DNGPrivateData provides a way for camera manufacturers to store private data in the DNG file for use by their own raw converters, and to have that data preserved by programs that edit DNG files.

The private data must follow these rules:

- *The private data must start with a null-terminated ASCII string identifying the data.* The first part of this string must be the manufacturer's name, to avoid conflicts between manufacturers.
- *The private data must be self-contained.* All offsets within the private data must be offsets relative to the start of the private data, and must not point to bytes outside the private data.
- *The private data must be byte-order independent.* If a DNG file is converted from a big-endian file to a little-endian file, the data must remain valid.

MakerNoteSafety

Tag	50741 (C635.H)
Type	SHORT
Count	1
Value	0 (unsafe) or 1 (safe)
Default	0
Usage	IFD 0

Description

MakerNoteSafety lets the DNG reader know whether the EXIF MakerNote tag is safe to preserve along with the rest of the EXIF data. File browsers and other image management software processing an image with a preserved MakerNote should be aware that any thumbnail image embedded in the MakerNote may be stale, and may not reflect the current state of the full size image.

A MakerNote is safe to preserve if it follows these rules:

- *The MakerNote data must be self-contained.* All offsets within the MakerNote must be offsets relative to the start of the MakerNote, and must not point to bytes outside the MakerNote.
- *The MakerNote data must be byte-order independent.* Moving the data to a file with a different byte order must not invalidate it.

RawDataUniqueID

Tag	50781 (C65D.H)
Type	BYTE
Count	16
Value	See below
Default	Optional
Usage	IFD 0

Description

This tag contains a 16-byte unique identifier for the raw image data in the DNG file. DNG readers can use this tag to recognize a particular raw image, even if the file's name or the metadata contained in the file has been changed.

If a DNG writer creates such an identifier, it should do so using an algorithm that will ensure that it is very unlikely two different images will end up having the same identifier.

OriginalRawFileName

Tag	50827 (C68B)
Type	ASCII or BYTE
Count	Byte count including null
Value	Null terminated UTF-8 encoded Unicode string
Default	Optional
Usage	IFD 0

Description

If the DNG file was converted from a non-DNG raw file, then this tag contains the file name of that original raw file.

OriginalRawFileData

Tag	50828 (C68C.H)
Type	UNDEFINED
Count	Byte count of embedded data
Value	See below
Default	Optional
Usage	IFD 0

Description

If the DNG file was converted from a non-DNG raw file, then this tag contains the compressed contents of that original raw file.

The contents of this tag always use the big-endian byte order.

The tag contains a sequence of data blocks. Future versions of the DNG specification may define additional data blocks, so DNG readers should ignore extra bytes when parsing this tag. DNG readers should also detect the case where data blocks are missing from the end of the sequence, and should assume a default value for all the missing blocks.

There are no padding or alignment bytes between data blocks. The sequence of data blocks is:

1. Compressed data fork of original raw file.
2. Compressed Mac OS resource fork of original raw file.
3. Mac OS file type (4 bytes) of original raw file.
4. Mac OS file creator (4 bytes) of original raw file.
5. Compressed data fork of sidecar ".THM" file.
6. Compressed Mac OS resource fork of sidecar ".THM" file.
7. Mac OS file type (4 bytes) of sidecar ".THM" file.
8. Mac OS file creator (4 bytes) of sidecar ".THM" file.

If the Mac OS file types or creator codes are unknown, zero is stored.

If the Mac OS resource forks do not exist, they should be encoded as zero length forks.

Each fork (data or Mac OS resource) is compressed and encoded as:

ForkLength = first four bytes. This is the uncompressed length of this fork. If this value is zero, then no more data is stored for this fork.

From ForkLength, compute the number of 64K compression blocks used for this data (the last block is usually smaller than 64K):

ForkBlocks = Floor ((ForkLength + 65535) / 65536)

The next (ForkBlocks + 1) 4-byte values are an index into the compressed data. The first ForkBlock values are offsets from the start of the data for this fork to the start of the compressed data for the corresponding compression block. The last value is an offset from the start of the data for this fork to the end of the data for this fork.

Following this index is the ZIP compressed data for each 64K compression block.

ActiveArea

Tag	50829 (C68D.H)
Type	SHORT or LONG
Count	4
Value	See below
Default	0, 0, ImageLength, ImageWidth
Usage	Raw IFD

Description

This rectangle defines the active (non-masked) pixels of the sensor. The order of the rectangle coordinates is: top, left, bottom, right.

MaskedAreas

Tag	50830 (C68E)
Type	SHORT or LONG
Count	4 * number of rectangles
Value	See below
Default	None
Usage	Raw IFD

Description

This tag contains a list of non-overlapping rectangle coordinates of fully masked pixels, which can be optionally used by DNG readers to measure the black encoding level.

The order of each rectangle's coordinates is: top, left, bottom, right.

If the raw image data has already had its black encoding level subtracted, then this tag should not be used, since the masked pixels are no longer useful.

Note that DNG writers are still required to include estimate and store the black encoding level using the black level DNG tags. Support for the MaskedAreas tag is not required of DNG readers.

AsShotICCProfile

Tag	50831 (C68F.H)
Type	UNDEFINED
Count	Length of ICC profile in bytes
Value	See below
Default	Optional
Usage	IFD 0

Description

This tag contains an ICC profile that, in conjunction with the AsShotPreProfileMatrix tag, provides the camera manufacturer with a way to specify a default color rendering from camera color space coordinates (linear reference values) into the ICC profile connection space.

The ICC profile connection space is an output referred colorimetric space, whereas the other color calibration tags in DNG specify a conversion into a scene referred colorimetric space. This means that the rendering in this profile should include any desired tone and gamut mapping needed to convert between scene referred values and output referred values.

DNG readers that have their own tone and gamut mapping controls (such as Adobe Camera Raw) will probably ignore this tag pair.

AsShotPreProfileMatrix

Tag	50832 (C690.H)
Type	SRATIONAL
Count	3 * ColorPlanes or ColorPlanes * ColorPlanes
Value	See below
Default	Identity matrix
Usage	IFD 0

Description

This tag is used in conjunction with the AsShotICCProfile tag. It specifies a matrix that should be applied to the camera color space coordinates before processing the values through the ICC profile specified in the AsShotICCProfile tag.

The matrix is stored in the row scan order.

If ColorPlanes is greater than three, then this matrix can (but is not required to) reduce the dimensionality of the color data down to three components, in which case the AsShotICCProfile should have three rather than ColorPlanes input components.

CurrentICCProfile

Tag	50833 (C691.H)
Type	UNDEFINED
Count	Length of ICC profile in bytes
Value	See below
Default	Optional
Usage	IFD 0

Description

This tag is used in conjunction with the CurrentPreProfileMatrix tag.

The CurrentICCProfile and CurrentPreProfileMatrix tags have the same purpose and usage as the AsShotICCProfile and AsShotPreProfileMatrix tag pair, except they are for use by raw file editors rather than camera manufacturers.

CurrentPreProfileMatrix

Tag	50834 (C692.H)
Type	SRATIONAL
Count	3 * ColorPlanes or ColorPlanes * ColorPlanes
Value	See below
Default	Identity matrix
Usage	IFD 0

Description

This tag is used in conjunction with the CurrentICCProfile tag.

The CurrentICCProfile and CurrentPreProfileMatrix tags have the same purpose and usage as the AsShotICCProfile and AsShotPreProfileMatrix tag pair, except they are for use by raw file editors rather than camera manufacturers.

1. Additional Tags for Version 1.2.0.0

The following tags have been added for the 1.2.0.0 version of this specification.

Additional Tags for Version 1.2.0.0

The following tags have been added for the 1.2.0.0 version of this specification.

ColorimetricReference

Tag	50879 (C6BF.H)
Type	SHORT
Count	1
Value	0 or 1
Default	0
Usage	IFD 0

Description

The DNG color model documents a transform between camera colors and CIE XYZ values. This tag describes the colorimetric reference for the CIE XYZ values.

0 = The XYZ values are scene-referred.

1 = The XYZ values are output-referred, using the ICC profile perceptual dynamic range.

This tag allows output-referred data to be stored in DNG files and still processed correctly by DNG readers.

CameraCalibrationSignature

Tag	50931 (C6F3.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Empty string
Usage	IFD 0

Description

A UTF-8 encoded string associated with the CameraCalibration1 and CameraCalibration2 tags. The CameraCalibration1 and CameraCalibration2 tags should only be used in the DNG color transform if the string stored in the CameraCalibrationSignature tag exactly matches the string stored in the ProfileCalibrationSignature tag for the selected camera profile.

ProfileCalibrationSignature

Tag	50932 (C6F4.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Empty string
Usage	IFD 0 or Camera Profile IFD

A UTF-8 encoded string associated with the camera profile tags. The CameraCalibration1 and CameraCalibration2 tags should only be used in the DNG color transfer if the string stored in the CameraCalibrationSignature tag exactly matches the string stored in the ProfileCalibrationSignature tag for the selected camera profile.

ExtraCameraProfiles

Tag	50933 (C6F5.H)
Type	LONG
Count	Number of extra camera profiles
Value	Offsets to Camera Profile IFDs
Default	Empty list
Usage	IFD 0

Description

A list of file offsets to extra Camera Profile IFDs. The format of a camera profile begins with a 16-bit byte order mark (MM or II) followed by a 16-bit "magic" number equal to 0x4352 (CR), a 32-bit IFD offset, and then a standard TIFF format IFD. All offsets are relative to the start of the byte order mark. Note that the primary camera profile tags should be stored in IFD 0, and the ExtraCameraProfiles tag should only be used if there is more than one camera profile stored in the DNG file.

AsShotProfileName

Tag	50934 (C6F6.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	IFD 0

Description

A UTF-8 encoded string containing the name of the "as shot" camera profile, if any.

NoiseReductionApplied

Tag	50935 (C6F7.H)
Type	RATIONAL
Count	1
Value	See below
Default	0/0
Usage	Raw IFD

Description

This tag indicates how much noise reduction has been applied to the raw data on a scale of 0.0 to 1.0. A 0.0 value indicates that no noise reduction has been applied. A 1.0 value indicates that the "ideal" amount of noise reduction has been applied, i.e. that the DNG reader should not apply additional noise reduction by default. A value of 0/0 indicates that this parameter is unknown.

ProfileName

Tag	50936 (C6F8.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

A UTF-8 encoded string containing the name of the camera profile. This tag is optional if there is only a single camera profile stored in the file but is required for all camera profiles if there is more than one camera profile stored in the file.

ProfileHueSatMapDims

Tag	50937 (C6F9.H)
Type	LONG
Count	3
Value	HueDivisions SaturationDivisions ValueDivisions
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

This tag specifies the number of input samples in each dimension of the hue/saturation/value mapping tables. The data for these tables are stored in ProfileHueSatMapData1 and ProfileHueSatMapData2 tags. Allowed values include the following:

- HueDivisions >= 1
- SaturationDivisions >= 2
- ValueDivisions >=1

The most common case has ValueDivisions equal to 1, so only hue and saturation are used as inputs to the mapping table.

ProfileHueSatMapData1

Tag	50938 (C6FA.H)
Type	FLOAT
Count	HueDivisions * SaturationDivisions * ValueDivisions * 3
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

This tag contains the data for the first hue/saturation/value mapping table. Each entry of the table contains three 32-bit IEEE floating-point values. The first entry is hue shift in degrees; the second entry is saturation scale factor; and the third entry is a value scale factor. The table entries are stored in the tag in nested loop order, with the value divisions in the outer loop, the hue divisions in the middle loop, and the saturation divisions in the inner loop. All zero input saturation entries are required to have a value scale factor of 1.0. The hue/saturation/value table application is described in detail in Chapter 6.

ProfileHueSatMapData2

Tag	50939 (C6FB.H)
Type	FLOAT
Count	HueDivisions * SaturationDivisions * ValueDivisions * 3
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

This tag contains the data for the second hue/saturation/value mapping table. Each entry of the table contains three 32-bit IEEE floating-point values. The first entry is hue shift in degrees;

the second entry is a saturation scale factor; and the third entry is a value scale factor. The table entries are stored in the tag in nested loop order, with the value divisions in the outer loop, the hue divisions in the middle loop, and the saturation divisions in the inner loop. All zero input saturation entries are required to have a value scale factor of 1.0. The hue/saturation/value table application is described in detail in Chapter 6.

ProfileToneCurve

Tag	50940 (C6FC.H)
Type	FLOAT
Count	Samples * 2
Value	See below
Default	None
Usage	IFD 0 or Camera Profile IFD

Description

This tag contains a default tone curve that can be applied while processing the image as a starting point for user adjustments. The curve is specified as a list of 32-bit IEEE floating-point value pairs in linear gamma. Each sample has an input value in the range of 0.0 to 1.0, and an output value in the range of 0.0 to 1.0. The first sample is required to be (0.0, 0.0), and the last sample is required to be (1.0, 1.0). Interpolated the curve using a cubic spline.

ProfileEmbedPolicy

Tag	50941 (C6FD.H)
Type	LONG
Count	1
Value	See below
Default	0
Usage	IFD 0 or Camera Profile IFD

Description

This tag contains information about the usage rules for the associated camera profile. The valid values and meanings are:

- 0 = “allow copying”. The camera profile can be used to process, or be embedded in, any DNG file. It can be copied from DNG files to other DNG files, or copied from DNG files and stored on the user’s system for use in processing or embedding in any DNG file. The camera profile may not be used to process non-DNG files.
- 1 = “embed if used”. This value applies the same rules as “allow copying”, except it does not allow copying the camera profile from a DNG file for use in processing any image other than the image in which it is embedded, unless the profile is already stored on the user’s system.
- 2 = “embed never”. This value only applies to profiles stored on a user’s system but not already embedded in DNG files. These stored profiles can be used to process images but cannot be embedded in files. If a camera profile is already embedded in a DNG file, then this value has the same restrictions as “embed if used”.
- 3 = “no restrictions”. The camera profile creator has not placed any restrictions on the use of the camera profile.

ProfileCopyright

Tag	50942 (C6FE.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	IFD 0 or Camera Profile IFD

Description

A UTF-8 encoded string containing the copyright information for the camera profile. This string always should be preserved along with the other camera profile tags.

ForwardMatrix1

Tag	50964 (C714.H)
Type	SRATIONAL
Count	3 * ColorPlanes
Value	See below
Default	Optional
Usage	IFD 0 or Camera Profile IFD

Description

This tag defines a matrix that maps white balanced camera colors to XYZ D50 colors. Application is described in detail in Chapter 6.

ForwardMatrix2

Tag	50965 (C715.H)
Type	SRATIONAL
Count	3 * ColorPlanes
Value	See below
Default	Optional
Usage	IFD 0 or Camera Profile IFD

Description

This tag defines a matrix that maps white balanced camera colors to XYZ D50 colors. Application is described in detail in Chapter 6.

PreviewApplicationName

Tag	50966 (C716.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	Preview IFD

Description

A UTF-8 encoded string containing the name of the application that created the preview stored in the IFD.

PreviewApplicationVersion

Tag	50967 (C717.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	Preview IFD

Description

A UTF-8 encoded string containing the version number of the application that created the preview stored in the IFD.

PreviewSettingsName

Tag	50968 (C718.H)
Type	ASCII or BYTE
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	Preview IFD

Description

A UTF-8 encoded string containing the name of the conversion settings (for example, snapshot name) used for the preview stored in the IFD.

PreviewSettingsDigest

Tag	50969 (C719.H)
Type	BYTE
Count	16
Value	See below
Default	Optional
Usage	Preview IFD

Description

A unique ID of the conversion settings (for example, MD5 digest) used to render the preview stored in the IFD.

PreviewColorSpace

Tag	50970 (C71A.H)
Type	LONG
Count	1
Value	See below
Default	See below
Usage	Preview IFD

Description

This tag specifies the color space in which the rendered preview in this IFD is stored. The valid values include:

- 0 = Unknown
- 1 = Gray Gamma 2.2
- 2 = sRGB
- 3 = Adobe RGB
- 4 = ProPhoto RGB

The default value for this tag is sRGB for color previews and Gray Gamma 2.2 for monochrome previews.

PreviewDateTime

Tag	50971 (C71B.H)
Type	ASCII
Count	Length of string including null
Value	Null terminated string
Default	Optional
Usage	Preview IFD

Description

This tag is an ASCII string containing the name of the date/time at which the preview stored in the IFD was rendered. The date/time is encoded using ISO 8601 format.

RawImageDigest

Tag	50972 (C71C.H)
Type	BYTE
Count	16
Value	See below
Default	Optional
Usage	IFD 0

Description

This tag is an MD5 digest of the raw image data. All pixels in the image are processed in row-scan order. Each pixel is zero padded to 16 or 32 bits deep (16-bit for data less than or equal to 16 bits deep, 32-bit otherwise). The data for each pixel is processed in little-endian byte order.

OriginalRawFileDigest

Tag	50973 (C71D.H)
Type	BYTE
Count	16
Value	See below
Default	Optional
Usage	IFD 0

Description

This tag is an MD5 digest of the data stored in the OriginalRawFileData tag.

SubTileBlockSize

Tag	50974 (C71E.H)
Type	SHORT or LONG
Count	2
Value	SubTileBlockRows SubTileBlockCols
Default	1, 1
Usage	Raw IFD

Description

Normally, the pixels within a tile are stored in simple row-scan order. This tag specifies that the pixels within a tile should be grouped first into rectangular blocks of the specified size. These blocks are stored in row-scan order. Within each block, the pixels are stored in row-scan order. The use of a non-default value for this tag requires setting the DNGBackwardVersion tag to at least 1.2.0.0.

RowInterleaveFactor

Tag	50975 (C71F.H)
Type	SHORT or LONG
Count	1
Value	RowInterleaveFactor
Default	1
Usage	Raw IFD

Description

This tag specifies that rows of the image are stored in interleaved order. The value of the tag specifies the number of interleaved fields. The use of a non-default value for this tag requires setting the DNGBackwardVersion tag to at least 1.2.0.0.

ProfileLookTableDims

Tag	50981 (C725.H)
Type	LONG
Count	3
Value	HueDivisions SaturationDivisions ValueDivisions
Default	none
Usage	IFD 0 or Camera Profile IFD

Description

This tag specifies the number of input samples in each dimension of a default "look" table. The data for this table is stored in the ProfileLookTableData tag. Allowed values include:

HueDivisions >= 1

SaturationDivisions >= 2

ValueDivisions >= 1

ProfileLookTableData

Tag	50982 (C726.H)
Type	FLOAT
Count	HueDivisions * SaturationDivisions * ValueDivisions * 3
Value	See below
Default	none
Usage	IFD 0 or Camera Profile IFD

Description

This tag contains a default "look" table that can be applied while processing the image as a starting point for user adjustment. This table uses the same format as the tables stored in the ProfileHueSatMapData1 and ProfileHueSatMapData2 tags, and is applied in the same color space. However, it should be applied later in the processing pipe, after any exposure compensation and/or fill light stages, but before any tone curve stage.

Each entry of the table contains three 32-bit IEEE floating-point values. The first entry is hue shift in degrees, the second entry is a saturation scale factor, and the third entry is a value scale factor.

The table entries are stored in the tag in nested loop order, with the value divisions in the outer loop, the hue divisions in the middle loop, and the saturation divisions in the inner loop.

All zero input saturation entries are required to have a value scale factor of 1.0.

Additional Tags for Version 1.3.0.0

The following tags have been added for the 1.3.0.0 version of this specification.

OpcodeList1

Tag	51008 (C740.H)
Type	UNDEFINED
Count	Variable
Value	Opcode List
Default	Empty List
Usage	Raw IFD

Description

Specifies the list of opcodes that should be applied to the raw image, as read directly from the file. The format and processing details of an opcode list are described in Chapter 7, "Opcode List Processing".

OpcodeList2

Tag	51009 (C741.H)
Type	UNDEFINED
Count	Variable
Value	Opcode List
Default	Empty List
Usage	Raw IFD

Description

Specifies the list of opcodes that should be applied to the raw image, just after it has been mapped to linear reference values. The format and processing details of an opcode list are described in Chapter 7, "Opcode List Processing".

OpcodeList3

Tag	51022 (C74E.H)
Type	UNDEFINED
Count	Variable
Value	Opcode List
Default	Empty List
Usage	Raw IFD

Description

Specifies the list of opcodes that should be applied to the raw image, just after it has been demosaiced. The format and processing details of an opcode list are described in Chapter 7, "Opcode List Processing".

NoiseProfile

Tag	51041 (C761.H)
Type	DOUBLE
Count	2 or 2 * ColorPlanes
Value	See Below
Default	Values are estimated from BaselineNoise tag (see below).
Usage	Raw IFD

Description

NoiseProfile describes the amount of noise in a raw image. Specifically, this tag models the amount of signal-dependent photon (shot) noise and signal-independent sensor readout noise, two common sources of noise in raw images. The model assumes that the noise is white and spatially independent, ignoring fixed pattern effects and other sources of noise (e.g., pixel response non-uniformity, spatially-dependent thermal effects, etc.).

This tag is intended to be used to describe the amount of noise present in unprocessed raw image data. When noise reduction has already been applied to the raw data (i.e., $\text{NoiseReductionApplied} > 0$), this tag may be used to estimate the white component of the residual noise.

For the purposes of this tag, noise is defined as the standard deviation of a random variable x, where x represents a recorded linear signal in the range $x \in [0,1]$. The two-parameter noise model is

$$N_i(x) = \sqrt{S_i x + O_i}$$

for parameters (S_i, O_i), where S_i is a scale term that models the amount of sensor amplification, and O_i is an offset term that models the amount of sensor readout noise. A more detailed explanation of this model is given below.

The data elements for this tag are the n sets of noise model parameters:

$$S_1, O_1, S_2, O_2, \ldots, S_n, O_n$$

Note that n must be 1 (i.e., tag count is 2) or equal to the number of color planes in the image (i.e., tag count is $2 \times \text{ColorPlanes}$). When $n = 1$, the two specified parameters (S_1, O_1) define the same noise model for all image planes. When n is equal to the number of image planes, the parameters (S_i, O_i) define the noise model for the i th image plane, e.g., (S_1, O_1) correspond to the first image plane, (S_2, O_2) correspond to the second image plane, etc. The order of the parameters follows the plane order specified by the CFAPlaneColor tag.

Each S_i term must be positive ($S_i > 0$), and each O_i term must be non-negative ($O_i \geq 0$).

A BaselineNoise tag value of 1.0 at ISO 100 corresponds approximately to NoiseProfile parameter values of $S_i = 2 \times 10^{-5}$ and $O_i = 4.5 \times 10^{-7}$ (e.g., standard deviation of ~ 0.00201 when $x = 0.18$); these values may be used to estimate absolute noise levels in an image when the NoiseProfile tag is missing. When both tags are present, however, DNG readers should prefer using the NoiseProfile data, since it describes noise levels more precisely than BaselineNoise.

A more detailed description of the noise model is given below. This tag models two common sources of noise:

1. Photon (shot) noise p, which has a white Poisson distribution, and
2. Electronic readout noise r, which is present even in the absence of light and is assumed to have an approximately white normal (Gaussian) distribution.

Assuming that p and r are independent random variables, the square of the total noise (i.e., the variance) can be expressed as the sum of the squares of the individual sources of noise:

$$N^2 = p^2 + r^2 \tag{1}$$

In this expression, the variables N, p, and r are expressed in B-bit recorded digital values, where common values of B include 12, 14, and 16 bits. If $\hat{x}$ is the average signal level expressed in photons, then its variance will also be $\hat{x}$, since a random variable with a Poisson distribution has a variance equal to its mean:

$$\hat{p}^2 = \hat{x} \tag{2}$$

where $\hat{p}$ denotes the photon noise, expressed in photons. The conversion factor between photons ($\hat{x}$, $\hat{p}$) and B-bit digital values (x, p) is the gain factor g:

$$x = g \cdot \hat{x} \tag{3}$$

$$p = g \cdot \hat{p} \tag{4}$$

Substituting equations 2, 3, and 4 into equation 1 yields:

$$\begin{aligned} N^2 &= (g \cdot \hat{p})^2 + r^2 \\ &= (g^2 \cdot \hat{x}) + r^2 \\ &= (g \cdot x) + r^2 \end{aligned}$$

Therefore the total noise N can be expressed as a two-parameter function of the signal x:

$$\begin{aligned} N(x) &= \sqrt{g \cdot x + r^2} \\ &= \sqrt{S_i s + O_i} \end{aligned}$$

for model parameters $S_i = g, O_i = r^2$.

This tag uses the convention of a normalized noise model, i.e., $N(x)$ is the standard deviation (i.e., noise) of a random variable x, where x represents a recorded linear signal in the range $x \in [0,1]$. The specified parameters (S_i, O_i) must also be appropriately normalized.

5 Mapping Raw Values to Linear Reference Values

The section describes DNG's processing model for mapping stored raw sensor values into linear reference values.

Linear reference values encode zero light as 0.0, and the maximum useful value (limited by either sensor saturation or analog to digital converter clipping) as 1.0. If SamplesPerPixel is greater than one, each sample plane should be processed independently.

The processing model follows these steps:

- Linearization
- Black Subtraction
- Rescaling
- Clipping

Linearization

The first step is to process the raw values through the look-up table specified by the LinearizationTable tag, if any. If the raw value is greater than the size of the table, it is mapped to the last entry of the table.

Black Subtraction

The black level for each pixel is then computed and subtracted. The black level for each pixel is the sum of the black levels specified by the BlackLevel, BlackLevelDeltaH and BlackLevelDeltaV tags.

Rescaling

The black subtracted values are then rescaled to map them to a logical 0.0 to 1.0 range. The scale factor is the inverse of the difference between the value specified in the WhiteLevel tag and the maximum computed black level for the sample plane.

Clipping

The rescaled values are then clipped to a 0.0 to 1.0 logical range.

6 Mapping Camera Color Space to CIE XYZ Space

This section describes the DNG processing model for mapping between the camera color space coordinates (linear reference values) and CIE XYZ (with a D50 white point).

Camera Calibration Matrices

DNG 1.2.0.0 and later supports different companies creating the camera calibration tags using different reference cameras.

When rendering a DNG file using a camera profile, it is important to know if the selected camera profile was designed using the same reference camera used to create the camera calibration tags. If so, then the camera calibration tags should be used. If not, then it is preferable to ignore the camera calibration tags and use identity matrices instead in order to minimize the worse case calibration mismatch error.

This matching is done by comparing the CameraCalibrationSignature tag and the ProfileCalibrationSignature tag for the selected camera profile. If they match, then use the camera calibration tags. If not, then use identity matrices.

One or Two Color Calibrations

DNG provides for one or two sets of color calibration tags, each set optimized for a different illuminant. If both sets of color calibration tags are included, then the raw converter should interpolate between the calibrations based on the white balance selected by the user.

If two calibrations are included, then it is recommended that one of the calibrations be for a low color temperature illuminant (e.g., Standard-A) and the second calibration illuminant be for a higher color temperature illuminant (e.g., D55 or D65). This combination has been found to work well for a wide range of real-world digital camera images.

DNG versions earlier than 1.2.0.0 allow the raw converter to choose the interpolation algorithm. DNG 1.2.0.0 and later requires a specific interpolation algorithm: linear interpolation using inverse correlated color temperature.

To find the interpolation weighting factor between the two tag sets, find the correlated color temperature for the user-selected white balance and the two calibration illuminants. If the white balance temperature is between two calibration illuminant temperatures, then invert all the temperatures and use linear interpolation. Otherwise, use the closest calibration tag set.

Definitions used in the following sections

Let n be the dimensionality of the camera color space (usually 3 or 4).

Let CM be the n-by-3 matrix interpolated from the ColorMatrix1 and ColorMatrix2 tags.

Let CC be the n-by-n matrix interpolated from the CameraCalibration1 and CameraCalibration2 tags (or identity matrices, if the signatures don't match).

Let AB be the n-by-n matrix, which is zero except for the diagonal entries, which are defined by the AnalogBalance tag.

Let RM be the 3-by-n matrix interpolated from the ReductionMatrix1 and ReductionMatrix2 tags.

Let FM be the 3-by-n matrix interpolated from the ForwardMatrix1 and ForwardMatrix2 tags.

Translating White Balance xy Coordinates to Camera Neutral Coordinates

If the white balance is specified in terms of a CIE xy coordinate, then a camera neutral coordinate can be derived by first finding the correlated color temperature for the xy value. This value determines the interpolation weighting factor between the two sets of color calibration tags.

The XYZ to camera space matrix is:

XYZtoCamera = AB * CC * CM

The camera neutral can be found by expanding the xy value to a 3-by-1 XYZ matrix (assuming Y = 1.0) and multiplying it by the XYZtoCamera matrix:

CameraNeutral = XYZtoCamera * XYZ

Translating Camera Neutral Coordinates to White Balance xy Coordinates

This process is slightly more complex than the transform in the other direction because it requires an iterative solution.

1. Guess an xy value. Use that guess to find the interpolation weighting factor between the color calibration tags. Find the XYZtoCamera matrix as above.

2. Find a new xy value by computing:

   XYZ = Inverse (XYZtoCamera) * CameraNeutral

   (If the XYZtoCamera matrix is not square, then use the pseudo inverse.)

3. Convert the resulting XYZ to a new xy value.

4. Iterate until the xy values converge to a solution.

Camera to XYZ (D50) Transform

DNG 1.2.0.0 and later support two methods of specifying the camera to XYZ (D50) transform, depending on whether or not the forward matrix tags are included in the camera profile.

The use of the forward matrix tags is recommended for two reasons. First, it allows the camera profile creator to control the chromatic adaptation algorithm used to convert between the calibration illuminant and D50. Second, it causes the white balance adjustment (if the user white balance does not match the calibration illuminant) to be done by scaling the camera coordinates rather than by adapting the resulting XYZ values, which has been found to work better in extreme cases.

If the ForwardMatrix tags are not included in the camera profile:

1. First, invert the XYZtoCamera matrix.

   If n = 3, this is:

   CameraToXYZ = Inverse (XYZtoCamera)

   If n > 3, and the reduction matrix tags are included, then:

   CameraToXYZ = Inverse (RM * XYZtoCamera) * RM

   Otherwise:

   CameraToXYZ = PseudoInverse (XYZtoCamera)

2. The white balanced transform is computed:

   CameraToXYZ_D50 = CA * CameraToXYZ

   CA, above, is a chromatic adaptation matrix that maps from the white balance xy value to the D50 white point. The recommended method for computing this chromatic adaptation matrix is to use the linear Bradford algorithm.

If the ForwardMatrix tags are included in the camera profile:

CameraToXYZ_D50 = FM * D * Inverse (AB * CC)

D, above, is a diagonal n-by-n matrix, computed so that the CameraToXYZ_D50 matrix maps the selected camera neutral to XYZ D50. The forward matrix is required to map a unit vector to XYZ D50 by definition, so D can be computed by finding the neutral for the reference camera:

ReferenceNeutral = Inverse (AB * CC) * CameraNeutral

And then: D = Invert (AsDiagonalMatrix (ReferenceNeutral))

Applying the Hue/Saturation/Value Mapping Table

After the camera colors have been converted to XYZ (D50) values, the Hue/Saturation/Value mapping table, if any, is applied. If there are two Hue/Saturation/Value mapping tables, then they are interpolated in the same way that color calibration tags are interpolated. If only one Hue/Saturation/Value table is included, then it is used regardless of the selected white balance.

1. First, the XYZ (D50) values are converted to linear RGB coordinates, using the ProPhoto RGB primaries. (This is also known as RIMM space).
2. The linear RGB coordinates are converted to HSV coordinates (Hue-Saturation-Value).
3. The HSV coordinates are used to index the mapping table using tri-linear interpolation, resulting in three values: hue shift (in degrees); saturation scale factor; value scale factor. If the division count in a dimension is 1, then the table is constant for that dimension.
4. Hue is indexed using “wrap-around” math. For example, if HueDivisions is equal to 3, then the table samples are at 0 degrees (red), 120 degrees (green), and 240 degrees (blue).
5. The hue coordinate is modified by adding the hue shift.
6. The saturation coordinate is modified by multiplying by the saturation scale factor, and then clipping to no more than 1.0.
7. The value coordinate is modified by multiplying by the value scale factor, and then clipping to no more than 1.0.
8. The HSV coordinates are converted to linear RGB coordinates, and then back to XYZ (D50) values.

It is recommended that these tables be limited to use a ValueDivisions equal to 1, so the table is only indexed by hue and saturation. In this way, all colors with the same hue and saturation, but with different values, map to the same new hue and saturation while preserving their value ratios.

7 Opcode List Processing

An opcode list is a list of opcodes, each of which does some specified image processing operation. Each opcode is performed in sequence.

Opcode lists are always stored in big-endian byte order, no matter what the file's main byte order is. This allows DNG utility programs to copy opcode lists from file to file, without needing to understand their detailed internal structure.

At the start of opcode list, there is a 32-bit unsigned integer count, which contains the number of opcodes in the list. This is followed by the data for each opcode.

Each opcode starts with a 32-bit unsigned integer, which contains the opcode ID. The opcode ID identifies the specific opcode. Documentation for each supported opcode ID in provided later in this chapter.

Next is a 32-bit unsigned integer, which contains the DNG specification version in which the opcode ID was defined. It is expected that new opcode IDs will be defined in future DNG specification versions. A DNG reader should never attempt to process an opcode with a version higher than DNG specification it was written to support.

Next is a 32-bit unsigned integer, which contains various flag bits. There are two defined flag bits. If bit 0 (the least significant bit) is set to 1, the opcode is considered optional, and the DNG reader may decide to not apply this opcode if it wishes, or it does not understand the opcode ID. If bit 1 (the second to least significant bit) is set to 1, the opcode can be skipped when doing "preview quality" processing, and only needs to be applied when doing "full quality" processing.

Next is a 32-bit unsigned integer, containing the number of bytes in a variable size parameter area for the opcode. The format of this variable size parameter area is dependent on the specific opcode ID, and is documented later in this chapter, along with each supported opcode ID.

When processing an opcode list, image values are clipped after the application of each opcode to the logical range of the image being modified. For OpcodeList1, this range is 0 to $2^{32} - 1$ for images with a bit depth greater than 16, otherwise 0 to $2^{16} - 1$. For OpcodeList2 and OpcodeList3, the logical range is 0.0 to 1.0.

The rest of this chapter contains information on each supported opcode ID.

WarpRectilinear

Opcode ID	1
DNG Version	1.3.0.0

Parameters

N – LONG

For each coefficient set $i \in 1, 2, \ldots, N$:

$k_{r_0, i}$ – DOUBLE

$k_{r_1, i}$ – DOUBLE

$k_{r_2, i}$ – DOUBLE

$k_{r_3, i}$ – DOUBLE

$k_{t_0, i}$ – DOUBLE

$k_{t_1, i}$ – DOUBLE

$\hat{c}_x$ – DOUBLE

$\hat{c}_y$ – DOUBLE

Description

This opcode applies a warp to an image and can be used to correct geometric distortion and lateral (transverse) chromatic aberration for rectilinear lenses. The warp function supports both radial and tangential distortion correction.

Let $K_i = \{k_{r_0, i}, k_{r_1, i}, k_{r_2, i}, k_{r_3, i}, k_{t_0, i}, k_{t_1, i}\}$ denote the ith coefficient set, where $i \in 1, 2, \ldots, \ddot{N}$.

Parameter N is the number of coefficient sets. N must be 1 or the total number of image planes. If $N = 1$, then a single set of warp coefficients (i.e., K_1) is applied to all image planes, i.e., all planes undergo the same transformation. If $N > 1$, coefficient set K_i is used to process the ith image plane, e.g., K_1 defines the warp function for the first image plane, K_2 defines the warp function for the second image plane, etc.

Parameters $K_i = \{k_{r_0, i}, k_{r_1, i}, k_{r_2, i}, k_{r_3, i}, k_{t_0, i}, k_{t_1, i}\}$ are the radial and tangential coefficients that define the warp function for the ith image plane; see below for implementation details and restrictions. Note that if $k_{r_0,\,i} = 1$ and the remaining terms are zero, then the warp function is the identity (i.e., no warp will be applied).

Parameters $(\hat{c}_x, \hat{c}_y)$ are the normalized x- and y-coordinates of the optical center, relative to the top-left pixel of the image. Example 1: specifying (0.5, 0.5) means that the optical center lies exactly at the image center. Example 2: specifying (1, 0) means that the optical center lies at the top-right pixel of the image.

Processing of this opcode is performed as follows.

Let $I'_i(x', y')$ be the pixel value of the i th plane of the original unwarped image at pixel position (x', y')—i.e., before opcode processing.

Let $I_i(x, y)$ be the pixel value of the i th plane of the warped image at pixel position (x, y)—i.e., after opcode processing.

For each pixel (x, y) of the i th plane of the warped image, compute:

$$I_i(x,y) \xleftarrow[\text{resample}]{} I'_i(x',y')$$

i.e., the pixel at position (x', y') in the original unwarped image is effectively moved to position (x, y) in the final warped image, where

$$\begin{aligned}
x' &= c_x + m(\Delta x_r + \Delta x_t) \\
y' &= c_y + m(\Delta y_r + \Delta y_t) \\
c_x &= x_0 + \hat{c}_x(x_1 - x_0) \\
c_y &= y_0 + \hat{c}_y(y_1 - y_0) \\
m_x &= \max(|x_0 - c_x|, |x_1 - c_x|) \\
m_y &= \max(|y_0 - c_y|, |y_1 - c_y|) \\
m &= \sqrt{m_x^2 + m_y^2} \\
r &= \sqrt{\Delta x^2 + \Delta y^2} \\
f &= k_{r_0,\,i} + k_{r_1,\,i} r^2 + k_{r_2,\,i} r^4 + k_{r_3,\,i} r^6 \\
\Delta x &= (x - c_x)/m \\
\Delta y &= (y - c_y)/m
\end{aligned}$$

[Continued from the previous page...]

$$\Delta x_r = f\Delta x$$

$$\Delta y_r = f\Delta y$$

$$\Delta x_t = k_{t_0,\, i}(2\Delta x\Delta y) + k_{t_1,\, i}(r^2 + 2\Delta x^2)$$

$$\Delta y_t = k_{t_1,\, i}(2\Delta x\Delta y) + k_{t_0,\, i}(r^2 + 2\Delta y^2)$$

(x_0, y_0) = pixel coordinates of the top-left pixel of the warped image

(x_1, y_1) = pixel coord. of the bottom-right pixel of the warped image

Notes and Restrictions

$(\Delta x_r, \Delta y_r)$ and $(\Delta x_t, \Delta y_t)$ are the radial and tangential warp components, respectively.

m is the Euclidean distance (in pixels) from the optical center to the farthest pixel in the warped image.

r is the normalized Euclidean distance $(\text{in}[0, 1])$ from the optical center to a given pixel in the warped image.

It is recommended that implementations use a suitable resampling kernel, such as a cubic spline.

This opcode can be used to correct lateral (transverse) chromatic aberration by specifying the appropriate coefficients for each image plane separately.

Each coefficient set K_i must satisfy the following constraints. Let $(x', y') = F(x, y)$ be the 2D warp function defined above. Let $x' = F_x(x, y)$ and $y' = F_y(x, y)$ be the x-component and y-component of $F(x, y)$, respectively. Let $w(r) = k_{r_0,\, i}r + k_{r_1,\, i}r^3 + k_{r_2,\, i}r^5 + k_{r_3,\, i}r^7$. The constraints are:

$F(x, y)$ must be invertible.

$Fx(x, y)$ must be an increasing function of x for all $x \in [x_0, x_1]$, i.e., $\partial F_x(x, y)/\partial x > 0$.

$Fy(x, y)$ must be an increasing function of y for all $y \in [y_0, y_1]$, i.e., $\partial F_y(x, y)/\partial y > 0$.

$w(r)$ must be an increasing function of r for all $r \in [0, 1]$, i.e., $w'(r) > 0$.

WarpFisheye

Opcode ID	2
DNG Version	1.3.0.0

Parameters

N – LONG

For each coefficient set $i \in 1, 2, \ldots, N$:

$k_{r_0, i}$ – DOUBLE

$k_{r_1, i}$ – DOUBLE

$k_{r_2, i}$ – DOUBLE

$k_{r_3, i}$ – DOUBLE

$\hat{c}_x$ – DOUBLE

$\hat{c}_y$ – DOUBLE

Description

This opcode applies a warp to an image and can be used to “unwrap” an image captured with a fisheye lens and map it instead to a perspective projection. It can also be used to correct geometric distortion and lateral (transverse) chromatic aberration for both fisheye and rectilinear lenses.

Let $K_i = \{k_{r_0, i}, k_{r_1, i}, k_{r_2, i}, k_{r_3, i}\}$ denote the ith coefficient set, where $i \in 1, 2, \ldots, N$.

Parameter N is the number of coefficient sets. N must be 1 or the total number of image planes. If $N = 1$, then a single set of warp coefficients (i.e., K_1) is applied to all image planes, i.e., all planes undergo the same transformation. If $N > 1$, coefficient set K_i is used to process the ith image plane, e.g., K_1 defines the warp function for the first image plane, K_2 defines the warp function for the second image plane, etc.

Parameters $K_i = \{k_{r_0, i}, k_{r_1, i}, k_{r_2, i}, k_{r_3, i}\}$ are the coefficients that define the warp function for the ith image plane; see below for implementation details and restrictions.

Parameters $(\hat{c}_x, \hat{c}_y)$ are the normalized x- and y-coordinates of the optical center, relative to the top-left pixel of the image. Example 1: specifying (0.5, 0.5) means that the optical center lies exactly at the image center. Example 2: specifying (1, 0) means that the optical center lies at the top-right pixel of the image.

Processing of this opcode is performed as follows.

Let $I'_i(x', y')$ be the pixel value of the i th plane of the original unwarped image at pixel position (x', y')—i.e., before opcode processing.

Let $I_i(x, y)$ be the pixel value of the i th plane of the warped image at pixel position (x, y)—i.e., after opcode processing.

For each pixel (x, y) of the i th plane of the warped image, compute:

$$I_i(x, y) \xleftarrow[\text{resample}]{} I'_i(x', y')$$

i.e., the pixel at position (x', y') in the original unwarped image is effectively moved to position (x, y) in the final warped image, where

$$\begin{aligned}
x' &= c_x + (m \cdot f \cdot \Delta x) \\
y' &= c_y + (m \cdot f \cdot \Delta y) \\
c_x &= x_0 + \hat{c}_x(x_1 - x_0) \\
c_y &= y_0 + \hat{c}_y(y_1 - y_0) \\
m_x &= \max(|x_0 - c_x|, |x_1 - c_x|) \\
m_y &= \max(|y_0 - c_y|, |y_1 - c_y|) \\
m &= \sqrt{m_x^2 + m_y^2} \\
r &= \sqrt{\Delta x^2 + \Delta y^2} \\
\theta &= \arctan(r) \\
f &= \frac{1}{r}(k_{r_0,\, i}\theta + k_{r_1,\, i}\theta^3 + k_{r_2,\, i}\theta^5 + k_{r_3,\, i}\theta^7) \\
\Delta x &= (x - c_x)/m \\
\Delta y &= (y - c_y)/m \\
(x_0, y_0) &= \text{pixel coordinates of the top-left pixel of the warped image} \\
(x_1, y_1) &= \text{pixel coord. of the bottom-right pixel of the warped image}
\end{aligned}$$

Notes and Restrictions

m is the Euclidean distance (in pixels) from the optical center to the farthest pixel in the warped image.

r is the normalized Euclidean distance $(\text{in}[0, 1])$ from the optical center to a given pixel in the warped image.

It is recommended that implementations use a suitable resampling kernel, such as a cubic spline.

This opcode can be used to correct lateral (transverse) chromatic aberration by specifying the appropriate coefficients for each image plane separately.

Each coefficient set K_i must satisfy the following constraints. Let $w(r) = k_{r_0,i}\theta + k_{r_1,i}\theta^3 + k_{r_2,i}\theta^5 + k_{r_3,i}\theta^7$, where $\theta = \arctan(r)$. $w(r)$ must be an increasing function of r for all $r \in [0, 1]$, i.e., $w'(r) > 0$.

FixVignetteRadial

Opcode ID	3
DNG Version	1.3.0.0

Parameters

k_0 – DOUBLE

k_1 – DOUBLE

k_2 – DOUBLE

k_3 – DOUBLE

k_4 – DOUBLE

$\hat{c}_x$ – DOUBLE

$\hat{c}_y$ – DOUBLE

Description

This opcode applies a gain function to an image and can be used to correct vignetting.

Parameters $(k_0, k_1, k_2, k_3, k_4)$ define a radially-symmetric gain function, explained below. Note that if all k_i terms are zero, then the gain function is the identity (i.e., no gain will be applied).

Parameters $(\hat{c}_x, \hat{c}_y)$ are the normalized x- and y-coordinates of the optical center, relative to the top-left pixel of the image. Example 1: specifying (0.5; 0.5) means that the optical center lies exactly at the image center. Example 2: specifying (1, 0) means that the optical center lies at the top-right pixel of the image.

Processing of this opcode is performed as follows.

Let $I'_i(x, y)$ be the uncorrected pixel value of the i th plane of the image at pixel position (x, y) (i.e., before opcode processing).

Let $I_i(x, y)$ be the corrected pixel value of the i th plane of the image at pixel position (x, y) (i.e., after opcode processing).

For each pixel (x, y) in the i th plane of the image, compute:

$$I_i(x, y) = g \cdot I'_i(x, y)$$

where

$$g = 1 + k_0 r^2 + k_1 r^4 + k_2 r^6 + k_3 r^8 + k_4 r^{10}$$
$$c_x = x_0 + \hat{c}_x(x_1 - x_0)$$
$$c_y = y_0 + \hat{c}_y(y_1 - y_0)$$
$$m_x = \max(|x_0 - c_x|, |x_1 - c_x|)$$
$$m_y = \max(|y_0 - c_y|, |y_1 - c_y|)$$
$$m = \sqrt{m_x^2 + m_y^2}$$
$$r = \frac{1}{m}\sqrt{(x - c_x)^2 + (y - c_y)^2}$$
$$(x_0, y_0) = \text{pixel coordinates of the top-left pixel of the image}$$
$$(x_1, y_1) = \text{pixel coordinates of the bottom-right pixel of the image}$$

Note that m represents the Euclidean distance (in pixels) from the optical center to the farthest pixel in the image, and r represents the normalized Euclidean distance $(\text{in}[0, 1])$ from the optical center to a given pixel.

FixBadPixelsConstant

Opcode ID	4
DNG Version	1.3.0.0

Parameters

Constant - LONG

BayerPhase - LONG

Description

This opcode patches (interpolates over) bad pixels in a Bayer pattern CFA image. The bad pixels are marked in the image by setting the bad pixels to a value of Constant.

If BayerPhase is 0, then the top-left pixel of the image is red pixel.

If BayerPhase is 1, then the top-left pixel of the image is green pixel in a green/red row.

If BayerPhase is 2, then the top-left pixel of the image is green pixel in a green/blue row.

If BayerPhase is 3, then the top-left pixel of the image is blue pixel.

FixBadPixelsList

Opcode ID	5
DNG Version	1.3.0.0

Parameters

BayerPhase - LONG

BadPointCount - LONG

BadRectCount - LONG

For Each BadPointCount:

BadPointRow - LONG

BadPointColumn - LONG

For Each BadRectCount:

BadRectTop - LONG

BadRectLeft - LONG

BadRectBottom - LONG

BadRectRight - LONG

Description

This opcode patches (interpolates over) bad pixels and rectangles in a Bayer pattern CFA image. The bad pixels and rectangles are specified as parameters for this opcode.

If BayerPhase is 0, then the top-left pixel of the image is red pixel.

If BayerPhase is 1, then the top-left pixel of the image is green pixel in a green/red row.

If BayerPhase is 2, then the top-left pixel of the image is green pixel in a green/blue row.

If BayerPhase is 3, then the top-left pixel of the image is blue pixel.

BadPointCount is the number of bad pixels to patch. BadPointRow and BadPointColumn specify the coordinates of the bad pixels to patch.

BadRectCount is the number of bad rectangles to patch. BadRectTop, BadRectLeft, BadRectBottom, and BadRectRight specify the coordinates of the bad rectangles to patch.

TrimBounds

Opcode ID	6
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Description

This opcode trims the image to the rectangle specified by Top, Left, Bottom, and Right.

MapTable

Opcode ID	7
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

TableSize - LONG

For Each TableSize:

TableEntry - SHORT

Description

This opcode maps a specified area and plane range of an image through a 16-bit LUT.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

TableSize specifies the number of entries in the 16-bit LUT. TableEntry specifies each table entry.

MapPolynomial

Opcode ID	8
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

Degree - LONG

For Each Value 0...Degree:

Coefficient - DOUBLE

Description

This opcode maps a specified area and plane range of an image through a polynomial function.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

The mapping function is a polynomial of degree Degree. The maximum allowed value for Degree is 8. The coefficients are stored in increasing order, starting with the zero degree coefficient (constant term).

GainMap

Opcode ID	9
DNG Version	1.3.0.0

Parameters

```
Top - LONG
Left - LONG
Bottom - LONG
Right - LONG
Plane - LONG
Planes - LONG
RowPitch - LONG
ColPitch - LONG
MapPointsV - LONG
MapPointsH - Long
MapSpacingV - DOUBLE
MapSpacingH - DOUBLE
MapOriginV - DOUBLE
MapOriginH - DOUBLE
MapPlanes - LONG
For Each MapPointsV
    For Each MapPointsH
        For Each MapPlanes
            MapGain - FLOAT
```

Description

This opcode multiplies a specified area and plane range of an image by a gain map.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

The gain map is a sub-sampled 2-D 32-bit floating-point image. MapPointsV is the number of samples in vertical direction. MapPointsH is the number of samples in the horizontal direction.

The gain map is not required to cover the entire image being modified. Inside the gain map bounds, values are interpolated using bi-linear interpolation. Outside the gain map bounds, values are replicated from the edges of the gain map.

The origin of the gain map relative to image being modified is specified by MapOriginV (vertical direction) and MapOriginH (horizontal direction), which are in relative coordinates, where 1.0 is equal to the height or width of the image being modified, respectively.

The spacing between gain map points is specified by MapSpacingV (vertical direction) and MapSpacingH (horizontal direction). Again these are in relative coordinates, where 1.0 is equal to the height or width of the image being modified, respectively.

MapPlanes specifies the number of planes in the gain map. If Planes > MapPlanes, the last gain map plane is used for any remaining planes being modified.

DeltaPerRow

Opcode ID	10
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

Count - LONG

For Each Count:

  Delta - FLOAT

Description

This opcode applies a per-row delta (constant offset) to a specified area and plane range of an image.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

Count is the number of deltas, and is required to match the number of affected rows in the specified area.

The delta to add to each affected row is specified by Delta.

DeltaPerColumn

Opcode ID	11
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

Count - LONG

For Each Count:

  Delta - FLOAT

Description

This opcode applies a per-column delta (constant offset) to a specified area and plane range of an image.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

Count is the number of deltas, and is required to match the number of affected columns in the specified area.

The delta to add to each affected column is specified by Delta.

ScalePerRow

Opcode ID	12
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

Count - LONG

For Each Count:

    Scale - FLOAT

Description

This opcode applies a per-row scale to a specified area and plane range of an image.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

Count is the number of scale values, and is required to match the number of affected rows in the specified area.

The scale to multiply each affected row by is specified by Scale.

ScalePerColumn

Opcode ID	13
DNG Version	1.3.0.0

Parameters

Top - LONG

Left - LONG

Bottom - LONG

Right - LONG

Plane - LONG

Planes - LONG

RowPitch - LONG

ColPitch - LONG

Count - LONG

For Each Count:

  Scale - FLOAT

Description

This opcode applies a per-column scale to a specified area and plane range of an image.

The bounds of the area of the image to be affected is specified by the Top, Left, Bottom, Right parameters. The first plane, and the number of planes, to be modified are specified by the Plane and Planes parameters. If RowPitch not equal to one, then only every RowPitch rows starting at the Top are affected. If ColPitch is not equal to one, then only every ColPitch columns starting at Left are affected.

Count is the number of scale values, and is required to match the number of affected columns in the specified area.

The scale to multiply each affected column by is specified by Scale.

Appendix A: Compatibility with Previous Versions

This appendix documents only the differences between this and previous versions of the DNG specification that are relevant to compatibility. Differences that are not relevant to compatibility (e.g., new optional tags that DNG readers are not required to support) are not documented in this appendix.

This information is useful in enabling DNG readers to correctly read DNG files with older version numbers. It also helps determine what version DNG writers can include in the DNGBackwardVersion tag.

The first version of the DNG speciation that was published was version 1.0.0.0.

Compatibility Issue 1: ActiveArea Tag

The ActiveArea tag was added to the DNG specification in version 1.1.0.0. Previous versions of the DNG specification do not support storing masked pixels.

DNG writers should set the DNGBackwardVersion to a minimum of 1.1.0.0 if the masked pixels are stored in the DNG file.

Compatibility Issue 2: 16-bit Lossless JPEG Encoding

The Lossless JPEG encoder/decoder used by Adobe applications to read and write DNG files before version 1.1.0.0 incorrectly deviated from the JPEG specification when dealing with 16-bit data. Since both the encoder and decoder deviated in the same way, no data was lost; however the data stream did not exactly match the data stream specified in the Lossless JPEG specification.

Because the vast majority of DNG 1.0.0.0 files using 16-bit Lossless JPEG encoding were created by Adobe applications, it is strongly recommended that software that reads or writes DNG files with version numbers less than 1.1.0.0 incorporate this deviation. Software that reads or writes DNG files with version 1.1.0.0 or later can safely assume that the Lossless JPEG stream is fully compliant with the Lossless JPEG specification.

Description of deviation

Lossless JPEG encodes the difference between a predicted value and the actual value for each pixel. With 16-bit data, these differences are computed modulo 16-bits, so the range of possible differences is -32768 to + 32767. Two values are stored for the difference. First the number of bits required to store the difference (encoded via a Huffman code), and then the actual difference.

Using the difference encoding scheme in the Lossless JPEG specification, only one difference value would take 16-bits to store: -32768. The Lossless JPEG specification special cases this difference bit length, and since there is only one possible difference value it does not bother to use any bits to store the actual difference.

In earlier versions of DNG the special case logic is not present, and the difference value for -32768 is stored in the compressed data stream as with all other difference bit lengths.

Compatibility Issue 3: SubTileBlockSize

The SubTileBlockSize tag was added to the DNG specification in version 1.2.0.0. Earlier versions of the DNG specification did not support this feature.

DNG writers should set the DNGBackwardVersion tag to a minimum of 1.2.0.0 if non-default values are used for this tag.

Compatibility Issue 4: RowInterleaveFactor

The RowInterleaveFactor tag was added to the DNG specification in version 1.2.0.0. Earlier versions of the DNG specification did not support this feature.

DNG writers should set the DNGBackwardVersion tag to a minimum of 1.2.0.0 if non-default values are used for this tag.

Compatibility Issue 5: PreviewColorSpace

The PreviewColorSpace tag was added to the DNG specification in version 1.2.0.0. Earlier versions of the DNG specification did not support this feature.

DNG writers should set the DNGBackwardVersion tag to a minimum of 1.2.0.0 if non-default values are used for this tag.

Compatibility Issue 6: CFALayout

CFALayout values 6 through 9 were added to the DNG specification in version 1.3.0.0. Earlier versions of the DNG specification did not support these CFALayout values.

DNG writers should set the DNGBackwardVersion tag to a minimum of 1.3.0.0 if CFALayout values 6 through 9 are used.

Compatibility Issue 7: Opcode Lists

The OpcodeList1, OpcodeList2, and OpcodeList3 tags, plus the initial set of supported Opcode IDs, were added to the DNG specification in version 1.3.0.0.

DNG writers should not set the DNGBackwardVersion to a version less than the DNG version for any opcode that is not marked optional.